Amendments to the Specification:

Please replace the paragraph beginning on page 2, line 14, with the following amended paragraph:

To overcome the drawback of slow writing processes to disks, a write-cache memory with fast access times for writing is typically used to cache data to be stored on the disk. ~~Caching means in~~In present data processing systems ~~that~~ the working data structure is first copied to a fast-access write-cache memory. The write-cache memory, hereinafter also called cache memory, is separate from the working memory and keeps data in memory before it is written to the disk. Saving a file therefore involves conversion of working data to serialised data and writing the serialised data from the working memory to the write-cache memory, which is done by the application, and finally writing the serialised data from the write-cache memory to an assigned disk space, which is done by the file system.

Please replace the paragraph beginning on page 5, line 3, with the following amended paragraph:

According to the method of the invention, at least one part of the memory space of the persistent-memory device is allocated to a file system. Memory allocation is the process of assigning on request a part of memory space, such as a number of blocks. Allocation of memory space to a file system is unusual in memory management methods known in the art, because the prior art has considered the file system responsible for managing only persistent secondary storage space like disk space. According to the ~~the~~ method of the invention, however, the file system can be assigned a part of the memory space of a persistent-memory device in addition to disk space. Thus, after the allocation step the storage space that is under the responsibility of the file system includes not only disk space, but also a part of the memory space of a persistent-memory device.

Please replace the paragraph beginning on page 7, line 3, with the following amended paragraph:

In a further embodiment the allocating step comprises a step of giving away to the file system the power of reading access to said first part of said memory space. This way the responsibility for the data contained in the first part of the memory space is completely handed over to the file system. Any reading access to the data will have to be managed by the file system after this step. This corresponds to a situation known from conventional data processing systems in which working data have been written to the disk. However, in the present embodiment the data may not have been written to the disk yet and is still ~~save~~ saved even in the case of a system failure. An application requesting access to these data does not see a difference to data stored conventionally on a disk. The present embodiment allows the application to have a fast access to the data via the file system after a restart, for instance in the case of a system breakdown. Thus, the present embodiment is advantageous for write-caching in a permanent memory. There is no need to perform a step of writing to a disk immediately after securing the data in the persistent memory by the allocation step. The step of writing to a disk may be postponed to a later stage without any risk of data loss. An application is sure that the working data has been secured just as in a conventional disk access step immediately after the successful performance of the allocation step of the present embodiment.

Please replace the paragraph beginning on page 10, line 6, with the following amended paragraph:

A further preferred embodiment of the write-caching method of the invention applies to situations where the first data has a working data structure that differs from a file structure. Not all working data structures necessarily differ from a file structure. This depends on the particular application. In general, however, ~~there~~ working data in the first part of the memory do not exhibit a linear structure according to the standards of the particular application. In this case the present embodiment of the write-caching method comprises a step of allocating a fourth part of the memory space to the application for an

executable file that is adapted to converting the first data into file data, a step of writing the executable file to the fourth part of the memory space, and a step of allocating the fourth part of the memory space to the file system.

Please replace the paragraph beginning on page 11, line 1, with the following amended paragraph:

The executable file for transformation need only once be handed over to the file system. After that, as long as the application is active, the file system may keep the transformation routine and the associated fourth memory space under its responsibility. Alternatively, it is also possible that the transformation routine is transferred to the file system with every write-caching. The ~~lattter~~ latter alternative will be advantageous if persistent memory space is sparse while the further alternative is to be preferred when the processing load due to the data storage is to be reduced. In both alternative cases, after said transforming step or after said finishing step, respectively, a step of deallocating the fourth part of the memory space to said memory manager is performed. This way, the memory manager can allocate the memory space to another application.

Please replace the paragraph beginning on page 11, line 26, with the following amended paragraph:

Initiating the transforming step ~~is by~~by the file system has the advantage that unnecessary transformation steps can be avoided. Often times applications have automatic routines of securing data after a predetermined time span calculated from the last securing action. If the application is not finished, the working-data copy in the first part of the memory space may simply be overwritten without having transformed the previous copy into file data. This way the processing load is further reduced, again making use of the persistent nature of the memory space that does not bear the risk of data loss.

Please replace the paragraph beginning on page 17, line 1, with the following amended paragraph:

FIG. 5 represents a later stage of this system. The file system has applied the conversion routine to the working data structure 50. A file data structure created this way in a memory space 60 that is allocated by the memory manager to the file system. The memory range 38 containing original working data blocks 40, 42, and 44 (cf. FIGS. 2 to 4) ~~has been~~is deallocated from the file system to the memory manager~~. For~~, since the ~~working data structure 39~~working data structure 38 ~~has been~~is secured at the point in time when the file system has the copy 50 of the working data structure 38 and the conversion routine 58. It is at that point that the file system confirms success of the saving process to the application. The memory manager is free to allocate this memory range to another application or to the file system. Obviously, a part of the memory range 38 has been allocated to the file system for storing the file data structure 60.